CHAPTER 20

Adding Tables

In This Chapter

What's new in HTML 4 tables

Defining a table

Adding table data

Grouping rows

Grouping columns

Spanning rows and columns

Nesting tables

Adding finishing touches

Although tables have their place in a Web page, they should not format text. This chapter focuses on using tables in their intended manner: to display tabular data. For most of the other elements covered in this book, style sheets provide formatting. The first version of cascading style sheets (CSS1) did not provide adequate support for tables, so for this purpose, the style-sheet model falls short. As we'll discuss in Chapter 25, the implementation of style sheets varies between browsers. Because style sheets supported by older browsers don't sufficiently support table formatting, tables may need to include formatting information in the table definition.

This chapter covers everything from creating a basic table to grouping cells or columns to spanning cells or columns, and contains the most complete explanation of tables created with all available HTML 4 elements. Chapter 30 discusses the implications of CSS1 and CSS2 on formatting tables.

Introducing the HTML Table Model

HTML 4 adds to the potential formatting of tables on your Web page. For the first time, you can group rows and columns for purposes of formatting. Even though you have to include formatting in your table definition under CSS1 (unlike in the rest of HTML 4), you can include it at the group level, rather than at the cell or row level—a truly powerful feature.

In addition, you can group columns. To understand this profound change, examine the HTML 3.2 table model. Tables are created from left to right, top to bottom. The following table is defined in this order: A1, A2, A3, A4, B1, B2, B3, B4, C1, C2, C3, D1, D2, D3.

A1	A2	A3	A4
B1	B2	B3	B4
C1	C2	C3	
D1	D2		D3

If all the cells in the fourth column (A4, B4, and D3) had the same background color, that formatting information would have to be defined three times, once in each cell definition. Grouping formatting for rows would be easier, because all the cells in a row are defined together, but if you wanted to define formatting for all the rows except the first row, you'd have to put the definitions into each row.

Using the COLGROUP element, you can group columns for formatting purposes. As a result, you can define all the cells in a column to look the same without inserting formatting information into each cell. This feature is especially useful in terms of maintenance. If you change the background color of your page from your style sheet and find that the background color of one column doesn't match, you can change the formatting information only for that COLGROUP and have it reflected in every COLGROUP table cell.

In addition, the following set of elements in HTML 4 affects the formatting of rows: THEAD, TFOOT, and TBODY. You can define one or more rows as the THEAD, one or more rows as the TFOOT, and the remaining rows take on the TBODY formatting.

Defining the Table

Create a table in HTML with the TABLE element. As with the UL, OL, and DL elements, however, this element alone won't achieve your desired results. You have to define table data in your table to see anything on your page.

Table <TABLE>

Start Tag: Required

Content: CAPTION
COL
COLGROUP
THEAD
TFOOT
TBODY
TR, TD

End Tag: Required

Attributes: `id`, `class`, `lang`, `dir`, `events`, `style`, `text`

`align`: deprecated; use style sheets

`cellpadding`: the amount of white space between the contents of a cell and its border

`cellspacing`: the amount of white space between cells and between cells and the outside borders of the table

`border`: the width in pixels of the frame around the table

`frame`: the side, if any, of the border of the table that is visible (see following choices)

`rules`: the rules, if any, that appear between cells (see following choices)

`width`: width of the table either in pixels or as a percentage of the window

`summary`: purpose of this table for speech-synthesizing browsers

The `frame` attribute takes one of the following values:

- **void.** No sides. This is the default value.
- **above.** The top side only.
- **below.** The bottom side only.
- **hsides.** The top and bottom sides only.
- **vsides.** The left and right sides only.
- **lhs.** The left-hand side only.
- **rhs.** The right-hand side only.
- **border or box.** All four sides.

The `rules` attribute takes one of the following values:

- **none.** No rules. This is the default value.
- **groups.** Rules appear between row groups and column groups only.
- **rows.** Rules appear between rows only.
- **cols.** Rules appear between columns only.
- **all.** Rules appear between all rows and columns.

The following examples combine these interesting formatting options:

Row 1
Row 2
Row 3

frame="lhs", rules="rows"

Column 1	Column 2	Column 3

frame="above", rules="cols"

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6

frame="void", rules="groups"

Adding Table Data

All data in a table goes into cells. A `cell` is created with the `TD` element or the `TH` element. Before you start defining cells, first set up rows into which the browser puts the cells. If you don't define rows to hold your cells, each cell will be its own row — meaning your table will be exactly one column wide. Create a row with the `TR` element.

Table Row `<TR>`

Start Tag: Required

Content: `TD`

`TH`

End Tag: Optional

Attributes: `id`, `class`, `lang`, `dir`, `events`, `title`, `style`

`align`: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

char: indicate the character used for character alignment; default is decimal point

charoff: character offset from left when using character alignment

The TR element can contain both cell data and cell headers. To differentiate the two items, the cell headers are formatted in bold by your browser. To define the formatting for your header cells yourself, use the class attribute and define a class for your headers in your style sheet.

Table Data <TD>

Start Tag: Required

Content: Cell data

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

colspan: number of columns this cell spans (defaults to 1)

rowspan: number of rows this cell spans (defaults to 1)

abbr: abbreviated version of the cell's content

axis: a comma-delimited list of category names; used for associating each cell with a category in speech-synthesizing browsers

headers: ID that associates a header cell with table data; can be used for speech-synthesizing browsers, for grouping columns, or for style sheets

nowrap: deprecated; avoid wrapping text within this cell; can make cells very wide

char: indicate the character to be used for character alignment; default is decimal point

charoff: character offset from left when using character alignment

bgcolor: deprecated; background color for this cell

width: deprecated; width of cell in pixels

height: deprecated; height of cell in pixels

Table Heading <TH>

Start Tag: Required

Content: Cell heading

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

colspan: number of columns this cell spans (defaults to 1)

rowspan: number of rows this cell spans (defaults to 1)

abbr: abbreviated version of the cell's content

axis: a comma-delimited list of category names; used for associating each cell with a category in speech-synthesizing browsers

headers: ID that associates a header cell with table data; can be used for speech-synthesizing browsers, for grouping columns, or for style sheets

scope: scope covered by header cell; valid values are row, column, rowgroup, colgroup

nowrap: deprecated; don't wrap text within this cell

width: deprecated; width of cell in pixels

height: deprecated; height of cell in pixels

After all the preceding definitions, you may be rethinking putting a table into your page to display tabular data. Don't get discouraged. The following simple example shows how to include a table on your page:

```
<TABLE border="1" frame="border" rules="all">
  <TR>
       <TH>
       <TH>Grams of Fat
       <TH>Grams of Protein
  <TR>
       <TH>Graham Crackers
       <TD align="center">1
       <TD align="center">0
  <TR>
       <TH>Raspberry Yogurt
       <TD align="center">2
       <TD align="center">9
  <TR>
       <TH>Turkey Breast
       <TD align="center">1
```

```
        <TD align="center">6
</TABLE>
```

This example generates a table that looks roughly like the following:

	Grams of Fat	Grams of Protein
Graham Crackers	1	0
Raspberry Yogurt	2	9
Turkey Breast	1	6

In reality, the borders crowd the text more than in this illustration. You can, of course, include cellpadding to keep the table from looking crowded, but this feature complicates the simple example.

The next example adds a few more features. This example uses the `align` attribute with the value set to "char," and the default alignment character is the decimal place. Internet Explorer 5.5 and Netscape 4.7 do not support character alignment, but this feature should be included in future versions. If you want to change the alignment character, add the `char` attribute and set it to the character you want to align.

```
<TABLE cellspacing="2" cellpadding="2" border="1" frame="box"
rules="cols">
<TR align="center">
  <TD>
  <TH>Chicago
  <TH>Washington, DC
  <TH>New York
  <TH>Charlottesville
<TR>
  <TH align="center">Breakfast
  <TD align="char">12.95
  <TD align="char">9.95
  <TD align="char">15.95
  <TD align="char">4.95
<TR>
  <TH align="center">Lunch
  <TD align="char">13.50
  <TD align="char">9.50
  <TD align="char">16.00
  <TD align="char">6.50
<TR>
  <TH align="center">Dinner
  <TD align="char">35.00
```

```
    <TD align="char">26.00
    <TD align="char">55.00
    <TD align="char">18.00
  <TR>
    <TH align="center">Transportation
    <TD align="char">12.00
    <TD align="char">15.00
    <TD align="char">22.00
    <TD align="char">9.00
  <TR>
    <TH align="center">Entertainment
    <TD align="char">45.00
    <TD align="char">45.00
    <TD align="char">75.00
    <TD align="char">20.00
  <TR>
    <TH align="center">Total
    <TD align="char">118.45
    <TD align="char">105.45
    <TD align="char">183.95
    <TD align="char">58.45
  </TABLE>
```

Figure 20-1 shows how the text renders in Netscape.

Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Go to: file://HTML Bible/table.html What's Related
Internet Lookup New&Cool
Document: Done

	Chicago	Washington, DC	New York	Charlottesville
Breakfast	12.95	9.95	15.95	4.95
Lunch	13.50	9.50	16.00	6.50
Dinner	35.00	26.00	55.00	18.00
Transportation	12.00	15.00	22.00	9.00
Entertainment	45.00	45.00	75.00	20.00
Total	118.45	105.45	183.95	58.45

Figure 20-1: Bordered Table in Netscape Communicator 4.7

Grouping Rows

HTML 4 enables you to group rows using the THEAD, TFOOT, and TBODY elements. When you create a table, you commonly have a header row (with cell headings), a number of rows with data, and then a footer row with totals, as in the preceding example.

If you use the THEAD, TFOOT, and TBODY elements, you must use every element, even if you don't need the entire set. Although you must use them, you can leave them empty. The order in which they should appear is THEAD, TFOOT, and TBODY. Why in this particular order? The browser needs to be able to load both the header and the footer information before it loads the body, which could be hundreds of rows. Down the road, you will be able to fit the entire table in one screen with the body scrolling, or the browser automatically putting a header and a footer on each page, when it prints a long table.

Table Header <THEAD>

Start Tag: Required

Content: TR

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style, char, charoff

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

char: specifies a single character within a text fragment to act as an alignment axis. The default value for this attribute is the decimal point character for the current language as set by lang.

charoff: specifies the offset to the first occurrence of the alignment character on each line.

Table Footer <TFOOT>

Start Tag: Required

Content: TR

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style, char, charoff

char: specifies a single character within a text fragment to act as an alignment axis. The default value for this attribute is the decimal point character for the current language as set by lang.

charoff: specifies the offset to the first occurrence of the alignment character on each line.

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

Table Body <TBODY>

Start Tag: Required

Content: TR

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style, char, charoff

char: specifies a single character within a text fragment to act as an alignment axis. The default value for this attribute is the decimal point character for the current language as set by lang.

charoff: specifies the offset to the first occurrence of the alignment character on each line.

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

These three elements improve the appearance of the code from the preceding costs-and-cities table. The table renders as follows in Internet Explorer 4 and newer, but won't work properly in any version of Netscape; Netscape doesn't support these elements yet.

```
<TABLE cellspacing="2" cellpadding="2" border="1" frames="box"
rules="cols">
<THEAD align="center" class="table_header1">
<TR align="center">
  <TD>
       <TH>Chicago
       <TH>Washington, DC
       <TH>New York
       <TH>Charlottesville
<TFOOT align="char" class="table_footer1">
<TR>
  <TH align="center">Total
       <TD>118.45
       <TD>105.45
       <TD>183.95
       <TD>58.45
<TBODY align="char" class="table_body1">
<TR>
       <TH align="center">Breakfast
       <TD>12.95
       <TD>9.95
       <TD>15.95
       <TD>4.95
<TR>
       <TH align="center">Lunch
```

```
        <TD>13.50
        <TD>9.50
        <TD>16.00
        <TD>6.50
<TR>
        <TH align="center">Dinner
        <TD>35.00
        <TD>26.00
        <TD>55.00
        <TD>18.00
<TR>
        <TH align="center">Transportation
        <TD>12.00
        <TD>15.00
        <TD>22.00
        <TD>9.00
<TR>
        <TH align="center">Entertainment
        <TD>45.00
        <TD>45.00
        <TD>75.00
        <TD>20.00
</TABLE>
```

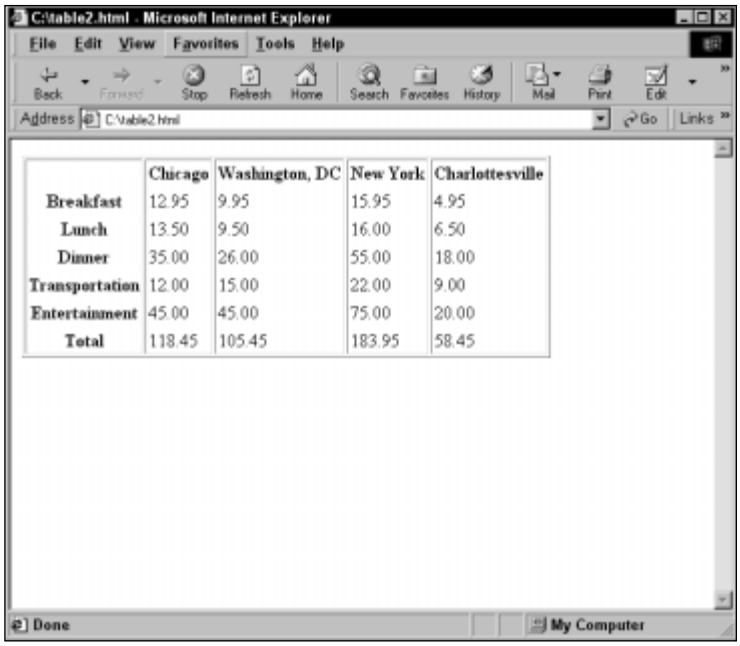
Figure 20-2 shows how the text renders in IE 5.0.

Several items in the preceding table definition may not have been clear in the first costs-and-cities-table example. First, notice the `align` attributes. These attributes are the only version of `align` that are not deprecated, because the text cannot be formatted in any other way within a cell. In addition, the first costs-and-cities table contains more instances of the `align` attribute. Grouping the rows enables you to move the `align="char"` attribute-value pair from each `TD` element to the `TBODY` element. Because the `TH` elements for each row still have an `align="center"` attribute-value pair, this condition overrides the `align` attribute definition in the `TBODY` element. The headings should be centered, rather than decimal-aligned.

All three of the new elements in this example have defined classes. As a result, classes must be defined in the style sheets for these elements. The classes are called `table_header1`, `table_footer1`, and `table_body1`, respectively.

The `align` attribute appears in the preceding examples with a value of "char." What does this mean? The "char" value causes alignment over a set character. The default is the decimal. You can change the default by using an additional attribute called `char`. If you have a set of cells that all have slashes (`/`), and you want to align the cells by the slash, use the following code:

```
align="char" char="/"
```


C:\table2.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address C:\table2.html
Go Links
Done
My Computer

	Chicago	Washington, DC	New York	Charlottesville
Breakfast	12.95	9.95	15.95	4.95
Lunch	13.50	9.50	16.00	6.50
Dinner	35.00	26.00	55.00	18.00
Transportation	12.00	15.00	22.00	9.00
Entertainment	45.00	45.00	75.00	20.00
Total	118.45	105.45	183.95	58.45

Figure 20-2: Bordered Table in IE 5.0

You can further customize the alignment of the cell values by using the `charoff` attribute with a length (the number of pixels by which to offset the alignment), as in the following example:

```
align="char" char="/" charoff="15"
```

Netscape Navigator and Internet Explorer do not currently support these uncommon usages.

Defining Columns

Where do you define a column? Until now, you haven't defined any columns. The browser calculates the number of columns based on the number of cells defined in a row. If you put more cells into one row than into other rows, the results can be unpredictable. The two major browsers do not fully support each column element.

Browsers prefer an up-front definition of the columns. You can perform this task with the `COL` element.

Column `<COL>`

Start Tag: Required

Content: TR

End Tag: Forbidden

Attributes: id, class, lang, dir, events, title, style, char, charoff

span: number of columns affected by COL attributes

char: specifies a single character within a text fragment to act as an alignment axis. The default value for this attribute is the decimal point character for the current language as set by lang.

charoff: specifies the offset to the first occurrence of the alignment character on each line.

width: column width

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

For example, suppose you want to define a table with five columns of 100 pixels each. Use the following COL element at the top of your table definition before you start defining rows or row groups:

```
<COL span="5" width="100">
```

If those five columns don't share the same width, use the following code:

```
<COL width="15%">
<COL width="25%">
<COL span="3" width="20%">
```

If you want the browser to set the widths, use either of the following code examples:

```
<COL width="0*" span="5">
```

or

```
<COL span="5">
```

Grouping Columns

You may need to use the COLGROUP element, which can be used with or without the COL element.

Column group <COLGROUP>

Start Tag: Required

Content: TR

End Tag: Optional

Attributes: id, class, lang, dir, events, title, style, char, charoff

span: number of columns included in group

char: specifies a single character within a text fragment to act as an alignment axis. The default value for this attribute is the decimal point character for the current language as set by lang.

charoff: specifies the offset to the first occurrence of the alignment character on each line.

width: column width

align: right, left, center, justify, char (these are not deprecated)

valign: top, middle, bottom, baseline (these are not deprecated)

If you define all a group's columns in the same manner, use the COLGROUP to both group columns and define them as in the following code. This example creates a table with six columns, with the first three in one group, the next two in another group, and the last one in a third group.

```
<COLGROUP span="3" style="group1">
<COLGROUP span="2" style="group2">
<COLGROUP span="1" style="group3">
```

If the first group has three columns of different widths, use the following code:

```
<COLGROUP span="3" style="group1">
  <COL width="75">
  <COL width="150">
  <COL width="100">
<COLGROUP span="2" style="group2">
<COLGROUP span="1" style="group3">
```

When you define the rest of your table, the fourth column will be formatted using style group2 without further instructions. The browser determines the fourth column in each row and applies the appropriate style.

Spanning Rows and Columns

Spanning rows and columns is easy, but neither major Version-4 browsers fully support all of these elements. To span columns, simply add an attribute called colspan to any cell you want to span more than one column. Because the default is one, set the value of colspan to any number greater than one to create a cell that is the specified number of columns wide.

To span rows, use the rowspan attribute, which works in the same way as colspan. Keep an eye on the arrangement of your cells, though. In the following representation, only one cell should be defined in both the first and third rows for a proper result.

	Chicago	Washington, DC	New York	Charlottesville
Breakfast	12.95	9.95	15.95	4.95
Lunch	13.50	9.50	16.00	6.50
Dinner	35.00	26.00	55.00	18.00
Transportation	12.00	15.00	22.00	9.00
Entertainment	45.00	45.00	75.00	20.00
Total	118.45	105.45	183.95	58.45

```
<TABLE border="1" frame="box" rules="all" cellspacing="2"
cellpadding="2">
<TR>
  <TD colspan="3">A1
<TR>
  <TD>B1
  <TD>B2
  <TD rowspan="2">B3
<TR>
  <TD colspan="2">C1
</TABLE>
```

Adding Finishing Touches to a Table

You can add two more elements to the table: a caption, created with the CAPTION element, and a summary, created with the summary attribute. Both features ease the comprehension of a table by speech-synthesizing browsers.

The CAPTION element must be defined immediately following the TABLE element start tag. The element's rendering depends on the particular browser.

Caption <CAPTION>

Start Tag: Required

Content: Caption to be displayed

End Tag: Required

Attributes: id, class, lang, dir, events, title, style align: deprecated; use style sheets

The `summary` attribute should be used as follows:

```
<TABLE summary="This table reflects the costs of living for a
day in major U.S. cities. It includes the costs of eating out
for breakfast, lunch, and dinner, the cost of transportation by
taxi, and the cost of entertainment in each city. Finally, it
shows totals for each city in the bottom row.">
<CAPTION>Daily Expenses for Travelers in Major U.S.
Cities</CAPTION>
```

Nesting Tables

Tables may be nested to produce interesting effects, such as the following example:

<table>
<tr><td colspan="3">A1</td></tr>
<tr><td>B1</td><td>B2</td><td rowspan="2">B3</td></tr>
<tr><td>C1</td><td></td></tr>
</table>

Because the `TABLE` element must have an end tag, no particular tricks exist for nesting tables. You may include a table within another table's cell or another table's row.

From Here

Jump to Chapter 33 to learn about formatting your page with CSS.

Proceed to Chapter 21 and learn creating forms and inserting scripts.

Summary

This chapter taught you everything you could possibly want to know about tables in HTML 4. You first discovered the table model in HTML. You also learned how to include defining tables as containers and defining rows and cells within tables.

You also learned how to nest tables and add features to improve table rendering by speech-synthesizing browsers. To make your tables more interesting, you learned how to specify column groups and row groups. To format your data more effectively, you learned how to span rows and columns.

✦ ✦ ✦